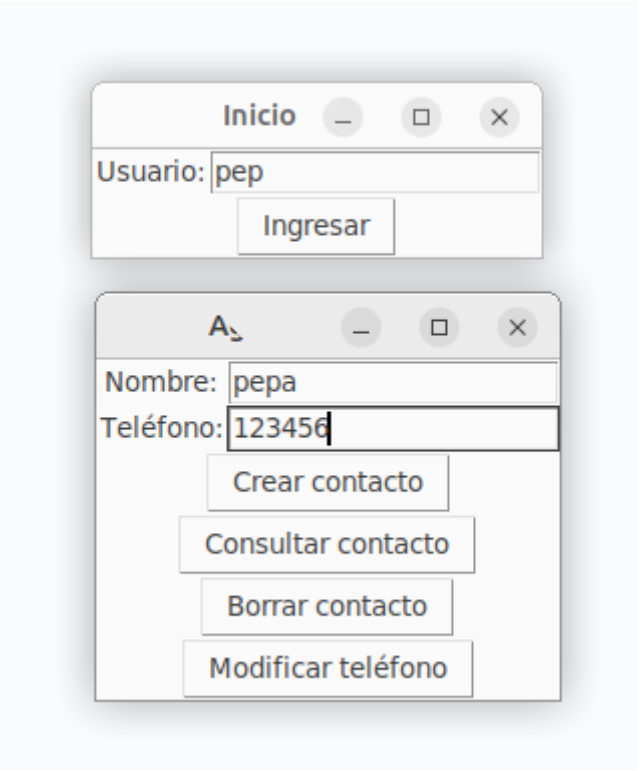


Títol del projecte: **Agenda de contactes.**

Nom: **Sofía Gràcia Mascarell.**

Data: **03/04/2024**

Link de GitHub:
https://github.com/SofiaGracia/AgendaContactes/tree/master

1. Índex

2. Introducció

El projecte "Agenda de contactes" es presenta com una ferramenta fonamental en l'àmbit de la gestió d'informació personal i professional. Desenvolupat amb l'objectiu de proporcionar una solució eficient i accessible per a l'organització de contactes, este projecte busca oferir una aplicació intuïtiva i versàtil que satisfaça les necessitats dels usuaris en termes d'emmagatzematge i gestió de contactes.

A través d'una interfície amigable i funcionalitats clau, com la creació, consulta, modificació i eliminació de contactes, l'Agenda de contactes busca simplificar el procés de mantindre i accedir a la informació de contacte rellevant. A més, la implementació d'una base de dades robusta garantix la persistència de les dades, permetent als usuaris accedir a la seua informació en qualsevol moment i des de qualsevol dispositiu.

Este projecte busca explorar i aplicar conceptes fonamentals de programació, disseny d'interfícies d'usuari i maneig de bases de dades. En resum, el projecte "Agenda de contactes" oferix una solució pràctica i educativa per a la gestió de contactes, brindant als usuaris una ferramenta útil i efectiva per a organitzar la seua xarxa de contactes de manera eficient i sistemàtica.

3. Ferramentes i mètodes

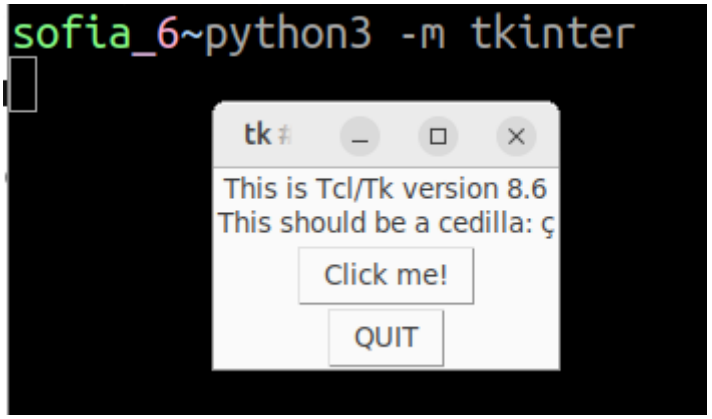
- **Descripció de les ferramentes i mètodes utilizades en el projecte:**

En primer lloc, es va optar per utilitzar **Visual Studio Code** com a entorn de desenvolupament integrat (IDE) a causa de les seues funcionalitats avançades i suport per a una àmplia gamma de llenguatges de programació. L'IDE s'utilitzà en un entorn **Ubuntu LTS** Jammy Jellyfish **24.04.3** que garantix l'estabilitat de l'entorn de desenvolupament i la compatibilitat amb les ferramentes utilitzades.

Per a la planificació i disseny del projecte, es va fer ús de la ferramenta **Umbrello**, que proporciona capacitats de modelatge visual per a la

representació de diagrames de classes, casos d'ús i altres artefactes de disseny. Esta ferramenta permet realitzar una organització prèvia i estructurada de les classes i casos d'ús que conformarien l'aplicació, facilitant així la comprensió i comunicació del disseny del programari.

Quant a les biblioteques i dependències utilitzades en el projecte, es va optar per integrar **biblioteques estàndard de Python** juntament amb aquelles específiques per al desenvolupament d'interfícies gràfiques d'usuari (GUI) i l'accés a bases de dades. Per a la interfície gràfica, es va fer ús de **Tkinter**, una biblioteca de GUI estàndard de Python (per a utilitzar a **Python3**)que oferix una àmplia gamma de ginys i ferramentes per a la creació d'interfícies intuïtives i funcionals.



Pel que fa a l'accés a la base de dades, es va utilitzar **SQLite**, una biblioteca lleugera i fàcil d'usar que proporciona un sistema de gestió de bases de dades relacional. A més, per a gestionar i explorar la base de dades SQLite, es va utilitzar **DB Browser for SQLite**, una ferramenta que proporciona una interfície gràfica fàcil d'usar per a administrar la base de dades a SQLite.

Per a fer el build i la distribucció de l'aplicació s'utilitzà **cx-Freeze.**

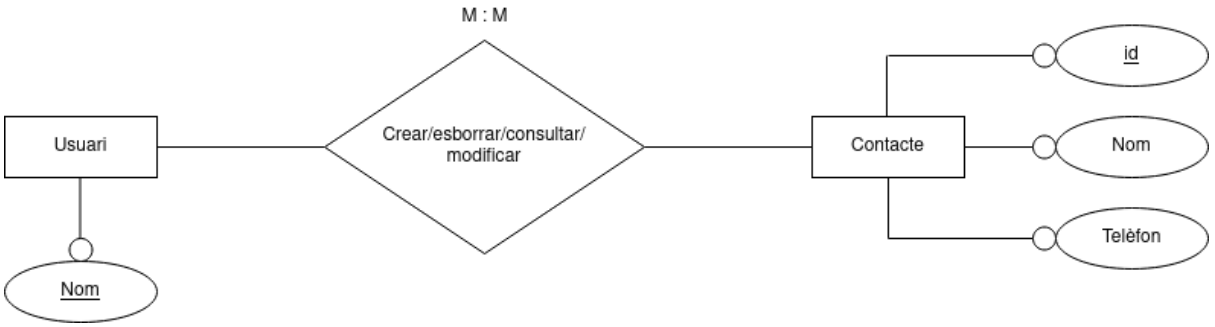
```
sofia_6~pip install cx-Freeze
Defaulting to user installation because normal site-packages is not writeable
Collecting cx-Freeze
  Downloading cx_Freeze-7.0.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_
x86_64.whl (13.4 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.5/13.4 MB 515.0 kB/s eta 0:00:22
```

Finalment, però no menys important, es va fer ús de **ChatGPT**, una ferramenta d'intel·ligència artificial desenvolupada per OpenAI, que va proporcionar assistència en la generació de text i la resolució de dubtes durant el procés de desenvolupament i documentació del projecte.

4. Perspectiva Estàtica

4.1. E/R (Entitat-Relación)

Diagrama que representa l'estructura de les entitats i les seues relacions:



4.2. Pas a taules

USUARI = nom

VNN: nom

CONTACTE = id + nom + telèfon

VNN: id, nom, telèfon

CREAR_CONTACTE = contacteID + usuari + nom + telèfon

C. Ali: contacteID -> CONTACTE (id)

usuari -> USUARI (nom)

4.3. DDL (Data Definition Language)

Exemples de codi que defineixen l'estructura de la base de dades: Creació de la taula de contactes. Dins de la classe AgendaApp, en el

mètode __init__(), s'executa una sentència SQL per crear la taula contactos si no existeix. El codi és el següent:

```
62          # Crear la tabla de contactos si no existe
63          self.cursor.execute('''CREATE TABLE IF NOT EXISTS contactos (
64                                     id INTEGER PRIMARY KEY AUTOINCREMENT,
65                                     nombre TEXT NOT NULL,
66                                     telefono TEXT NOT NULL,
67                                     usuario TEXT NOT NULL
68                                 )''')
69          self.conexion.commit()
```

4.4. DML (Data Manipulation Language)

Exemples de codi que manipulen les dades en la base de dades.

4.4.1 Inserció d'un nou contacte

En el mètode **crear_contacto()** de la classe AgendaApp, s'insereix un nou registre a la taula contactos amb el nom, telèfon i usuari proporcionats. El codi és el següent:

```
81          # Insertar un nuevo contacto en la base de datos para este usuario
82          self.cursor.execute("INSERT INTO contactos (nombre, telefono, usuario) VALUES (?, ?, ?)",
83                              (nombre, telefono, self.nombre_usuario))
84          self.conexion.commit()
85          messagebox.showinfo("Éxito", "Se ha creado el contacto")
```

4.4.2. Consulta d'un contacte per nom

En el mètode **consultar_contacto()** de la classe AgendaApp, s'executa una consulta SQL per buscar un contacte pel seu nom i usuari associat. El codi és el següent:

4.4.3. Borrar un contacte

En el mètode **borrar_contacto()** de la classe AgendaApp, s'executa una sentència SQL per eliminar un registre de la taula contactos segons el nom i usuari associat. El codi és el següent:

```
104         # Borrar el contacto y su teléfono correspondiente para este usuario
105         self.cursor.execute("DELETE FROM contactos WHERE nombre=? AND usuario=?",
106                             (nombre, self.nombre_usuario))
107         self.conexion.commit()
108         messagebox.showinfo("Éxito", "Se ha borrado el contacto y su teléfono correspondiente")
```

4.4.4. Modificació del telèfon d'un contacte

En el mètode **modificar_telefono()** de la classe AgendaApp, s'executa una sentència SQL per actualitzar el telèfon d'un contacte existent a la taula contactos:

```
# Modificar el teléfono del contacto existente para este usuario
self.cursor.execute("UPDATE contactos SET telefono=? WHERE nombre=? AND usuario=?",
                    (nuevo_telefono, nombre, self.nombre_usuario))
self.conexion.commit()
messagebox.showinfo("Éxito", "Se ha modificado el teléfono del contacto")
```

4.5. DQL (Data Query Language)

4.5.1. Consulta d'un contacte per nom

En el mètode **consultar_contacto()** de la classe AgendaApp, s'executa una consulta SQL per buscar un contacte pel seu nom i usuari associat. El codi és el següent:

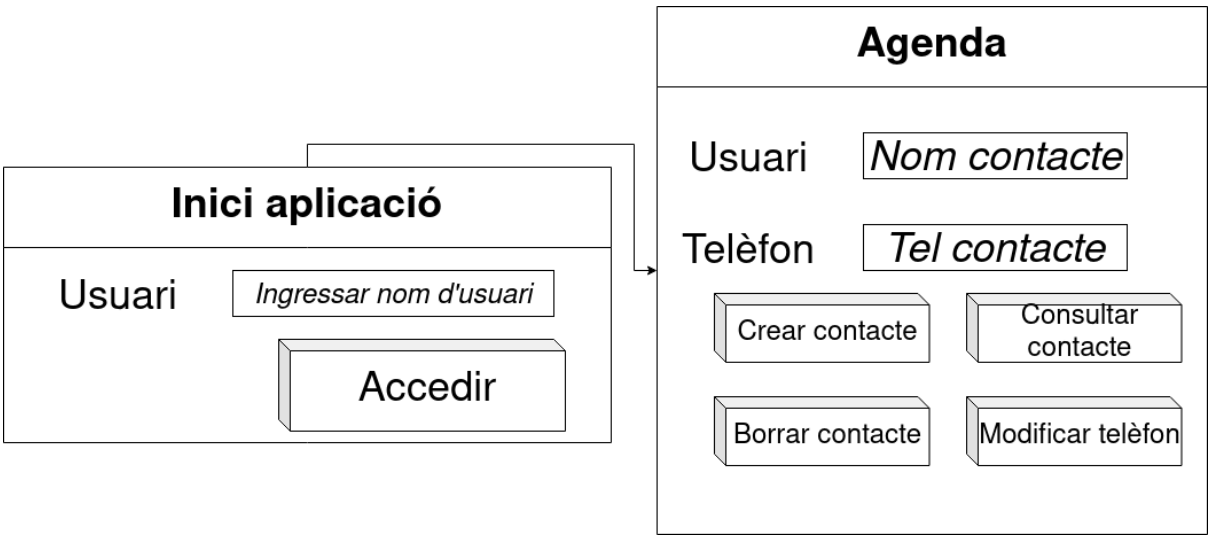
```
86      def consultar_contacto(self):
87          nombre = self.entry_nombre.get()
88
89          # Consultar un contacto por nombre para este usuario
90          self.cursor.execute("SELECT * FROM contactos WHERE nombre=? AND usuario=?",
91                              (nombre, self.nombre_usuario))
92          contacto = self.cursor.fetchone()
```

4.6. DCL (Data Control Language)

El codi proporcionat no inclou un control de permisos o seguretat específics de la base de dades. No obstant això, s'implementa una validació per verificar si un contacte existeix abans de realitzar operacions com ara esborrar-lo o modificar el seu telèfon. Això ajuda a evitar operacions no desitjades sobre la base de dades.

5. Perspectiva Dinámica

5.1. Sketch



5.2. Casos d'ús (Mètodes)

Descripció dels casos d'ús i com s'implementen.

Cas d'ús	Ingressar a l'aplicació.	**Identificador:** 1
Actors	Usuari	
Tipus	Primari	
Referències	Per executar l'aplicació necessitarem: - Python - Mòdul Tkinter - SQLite	
Precondició	Executar l'aplicació	
Postcondició	Ingresar el nom de l'usuari a utilitzar l'aplicació	
Descripció	L'usuari pot iniciar sessió en l'aplicació donant el seu nom	
Resum	Si l'usuari ingressa un nom d'usuari vàlid, s'obrirà la finestra de l'agenda. Si l'usuari intenta ingressar sense proporcionar un nom d'usuari, l'aplicació mostrarà un missatge d'error i sol·licitarà a l'usuari que ingresse un nom d'usuari vàlid.	

Curs normal 1: (Actor: Usuari)

1. L'usuari obri l'aplicació.
2. L'usuari ingressa el seu nom d'usuari.
3. L'usuari fa clic en el botó "Ingressar".
4. El sistema verifica que s'haja ingressat un nom d'usuari.
5. Si el nom d'usuari és vàlid, s'obri la finestra de l'agenda.

Curs altern 1 : (Actor: Usuari)

1. L'usuari obri l'aplicació.
2. L'usuari no ingressa el seu nom d'usuari.
3. L'usuari fa clic en el botó "Ingressar".
4. El sistema verifica que s'haja ingressat un nom d'usuari.
5. Es mostra el missatge d'error.

Cas d'ús	Crear contacto	**Identificador:** 2
Actors	Usuari	
Tipus	Primari	
Referències	Per executar l'aplicació necessitarem: - Python - Mòdul Tkinter - SQLite	
Precondició	- Ingressar a l'aplicació - Introduir el nom i el telèfon del contacte.	
Postcondició	Prémer opció per a gestionar contactes o tancar agenda.	
Descripció	Permet a l'usuari realitzar accions relacionades amb la gestió de contactes, com crear, consultar, modificar i eliminar contactes.	
Resum	L'opció permet a l'usuari registrar nom i telèfon d'un contacte. Si l'usuari intenta crear un contacte sense proporcionar informació obligatòria, com el nom, l'aplicació mostra un missatge d'error i sol·licita la informació que manca.	

Curs normal 2: (Actor: Usuari)
1. L'usuari obri la finestra de l'agenda.
2. L'usuari proporciona la informació necessària per a crear el contacte (nom i telèfon).
3. El sistema realitza l'acció corresponent i es registren les dades en la base de dades.
5. El sistema mostra un missatge indicant el resultat de l'acció.

Curs altern 2.1 : No es proporciona les dades (Actor: Usuari)

1. L'usuari obri la finestra de l'agenda.
2. L'usuari no proporciona nom i teléfon.
3.L'usuari selecciona una acció entre crear, consultar, modificar o eliminar contacte.
4. El sistema mostra el missatge d'error corresponent resultant de l'acció fallida.

Curs altern 2.2 : Es proporcionen dades existents (Actor: Usuari)

1. L'usuari obri la finestra de l'agenda.

2. L'usuari no proporciona nom i teléfon.
3.L'usuari selecciona l'opció de crear contacte.
4. El sistema mostra el missatge de que ja existeix el contacte.

Cas d'ús	Consultar contacte	**Identificador:** 3
Actors	Usuari	
Tipus	Primari	
Referències	Per executar l'aplicació necessitarem: - Python - Mòdul Tkinter - SQLite	
Precondició	- Ingressar a l'aplicació - Introduir el nom del contacte.	
Postcondició	Prémer opció per a gestionar contactes o tancar agenda.	
Descripció	Permet a l'usuari realitzar accions relacionades amb la gestió de contactes, com crear, consultar, modificar i eliminar contactes.	
Resum	L'opció permet consultar el telèfon d'un contacte a partir del nom introduit.	

Curs normal 3: (Actor: Usuari)
1. L'usuari obri la finestra de l'agenda.
2. L'usuari proporciona el nom del contacte..
3. El sistema mostra el número de telèfon del contacte.

Curs altern 3.1 : No es proporciona les dades (Actor: Usuari)

1. L'usuari obri la finestra de l'agenda.
2. L'usuari no proporciona nom.
4. El sistema mostra el missatge d'error corresponent resultant de l'acció fallida.

Curs altern 3.2 : Es proporciones dades inexistents (Actor: Usuari)

1. L'usuari obri la finestra de l'agenda.
2. L'usuari proporciona el nom d'un contacte que no existeix.
4. El sistema mostra el missatge d'error corresponent resultant de l'acció fallida.

Cas d'ús	Borrar contacte	**Identificador:** 4
Actors	Usuari	
Tipus	Primari	
Referències	Per executar l'aplicació necessitarem: - Python - Mòdul Tkinter	

	- SQLite
Precondició	Ingressar a l'aplicació
Postcondició	Prémer opció per a gestionar contactes o tancar agenda.
Descripció	Permet a l'usuari realitzar accions relacionades amb la gestió de contactes, com crear, consultar, modificar i eliminar contactes.
Resum	L'opció permet borrar el contacte i el seu telèfon de la base de dades.

Curs normal 4: (Actor: Usuari)
1. L'usuari obri la finestra de l'agenda.
2. L'usuari proporciona el nom del contacte.
3.L'usuari selecciona "borrrar contacte".
4. El sistema realitza l'acció corresponent en la base de dades.
5. El sistema mostra un missatge indicant el resultat de l'acció.

Curs altern 4.1 : No es proporciona les dades (Actor: Usuari)

1. L'usuari obri la finestra de l'agenda.
2. L'usuari no proporciona nom i teléfon.
3.L'usuari selecciona l'opció "borrar contacte".
4. El sistema mostra el missatge d'error corresponent resultant de l'acció fallida.

Curs altern 4.2 : Es proporciones dades inexistents (Actor: Usuari)

1. L'usuari obri la finestra de l'agenda.
2. L'usuari proporciona el nom d'un contacte que no existeix.
4. El sistema mostra el missatge d'error corresponent resultant de l'acció fallida.

Cas d'ús	Modificar contacto	**Identificador:** 5
Actors	Usuari	
Tipus	Primari	
Referències	Per executar l'aplicació necessitarem: - Python - Mòdul Tkinter - SQLite	
Precondició	Ingressar a l'aplicació	
Postcondició	Prémer opció per a gestionar contactes o tancar agenda.	
Descripció	Permet a l'usuari realitzar accions relacionades amb la gestió de contactes, com crear, consultar, modificar i eliminar contactes.	
Resum	L'opció permet modificar el telèfon del contacte de la base de dades.	

Curs normal 5: (Actor: Usuari)

1. L'usuari obri la finestra de l'agenda.
2. L'usuari proporciona el nom i el nou telèfon del contacte.
3.L'usuari selecciona l'acció de modificar el contacte.
4. El sistema realitza l'acció corresponent en la base de dades.
5. El sistema mostra un missatge indicant el resultat de l'acció.

Curs altern 5.1 : No es proporciona les dades (Actor: Usuari)

1. L'usuari obri la finestra de l'agenda.
2. L'usuari no proporciona nom i teléfon.
3.L'usuari selecciona l'acció de modificar el contacte.
4. El sistema mostra el missatge d'error corresponent resultant de l'acció fallida.

Cas d'ús	Sortir de l'aplicació	**Identificador:** 6
Actors	Usuari	
Tipus	Primari	
Referències	Per executar l'aplicació necessitarem: - Python - Mòdul Tkinter - SQLite	
Precondició	Ingressar a l'aplicació.	
Postcondició	L'aplicació es tanca correctament	
Descripció	Permet a l'usuari tancar l'aplicació de manera segura.	
Resum	No s'identifiquen excepcions específiques en este cas d'ús, ja que el tancament de l'aplicació es considera una acció estàndard i no està subjecte a condicions especials.	

Curs normal 6: (Actor: Usuari)

1. L'usuari fa clic en el botó de tancar finestra.
2. El sistema tanca la connexió a la base de dades.
3. El sistema tanca la finestra i finalitza l'aplicació.

Curs altern 6.1: (Actor: Usuari)

1. L'usuari no fa clic en el botó de tancar finestra.
2. El sistema no tanca la connexió a la base de dades.
3. No finalitza l'aplicació.

6. Conclusions

En resum, el projecte ha demostrat la importància d'explorar i aprofitar les biblioteques i ferramentes disponibles en el desenvolupament de programari. Si bé l'aplicació desenvolupada no és perfecta, ha sigut una oportunitat valuosa per a aplicar i combinar coneixements en programació i bases de dades.

A través de l'experiència pràctica d'este projecte, s'ha evidenciat que la incorporació de biblioteques específiques pot millorar significativament la funcionalitat i eficiència de les aplicacions desenvolupades. En este cas, l'ús de biblioteques com Tkinter per a la interfície gràfica i SQLite per a la gestió de la base de dades ha permés crear una aplicació funcional.

Este projecte és només el començament d'un viatge d'aprenentatge continu en el camp de la informàtica, on la pràctica i l'experimentació són clau per al creixement professional.

7. Webgrafia

Webgrafía:

1. Documentació oficial de Python: https://docs.python.org/
2. Documentació oficial de Tkinter: https://docs.python.org/3/library/tkinter.html
3. Documentació oficial de SQLite: https://sqlite.org/docs.html
4. Tutorial de Visual Studio Code: https://code.visualstudio.com/docs/introvideos/basics
5. Umbrello UML Modeller: https://umbrello.kde.org/
6. DB Browser for SQLite: https://sqlitebrowser.org/
7. Manual MySQL: http://dev.mysql.com/doc/refman/5.0/en/